



# MYFLIX CASE STUDY

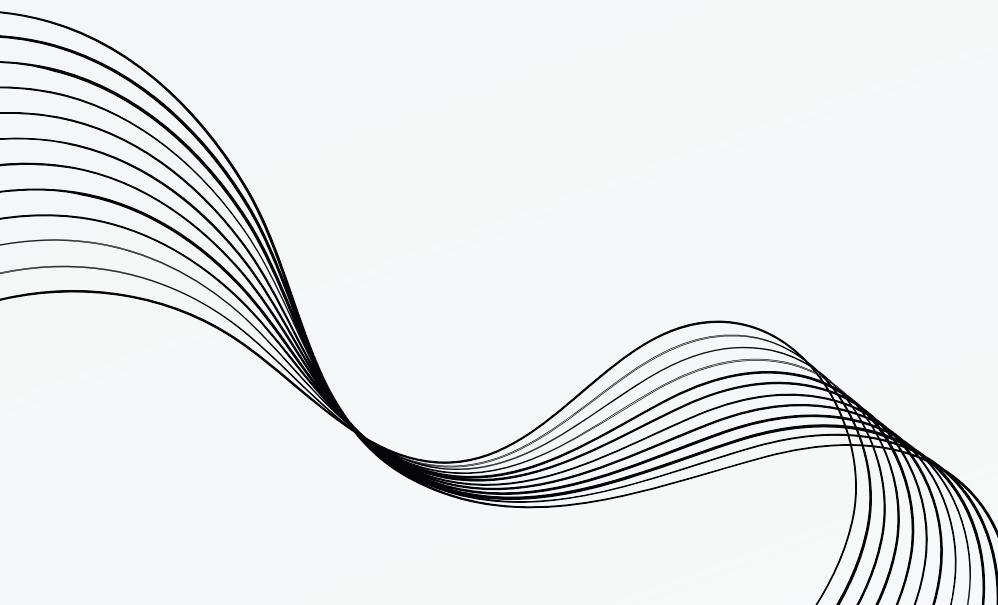
# CONTENT

- 
- 01** ABOUT
  - 02** PROBLEM STATEMENT
  - 03** DEVELOPMENT PHASES
  - 04** SERVER-SIDE DEVELOPMENT
  - 05** CLIENT-SIDE DEVELOPMENT (REACT & ANGULAR)
  - 06** SUMMARY AND CONCLUSION
  - 07** STATISTICS, TESTIMONIALS & LIVE DEMOS

# ABOUT



Welcome to the MyFlix case study. This presentation will walk you through the development process of a comprehensive web application designed for movie enthusiasts.



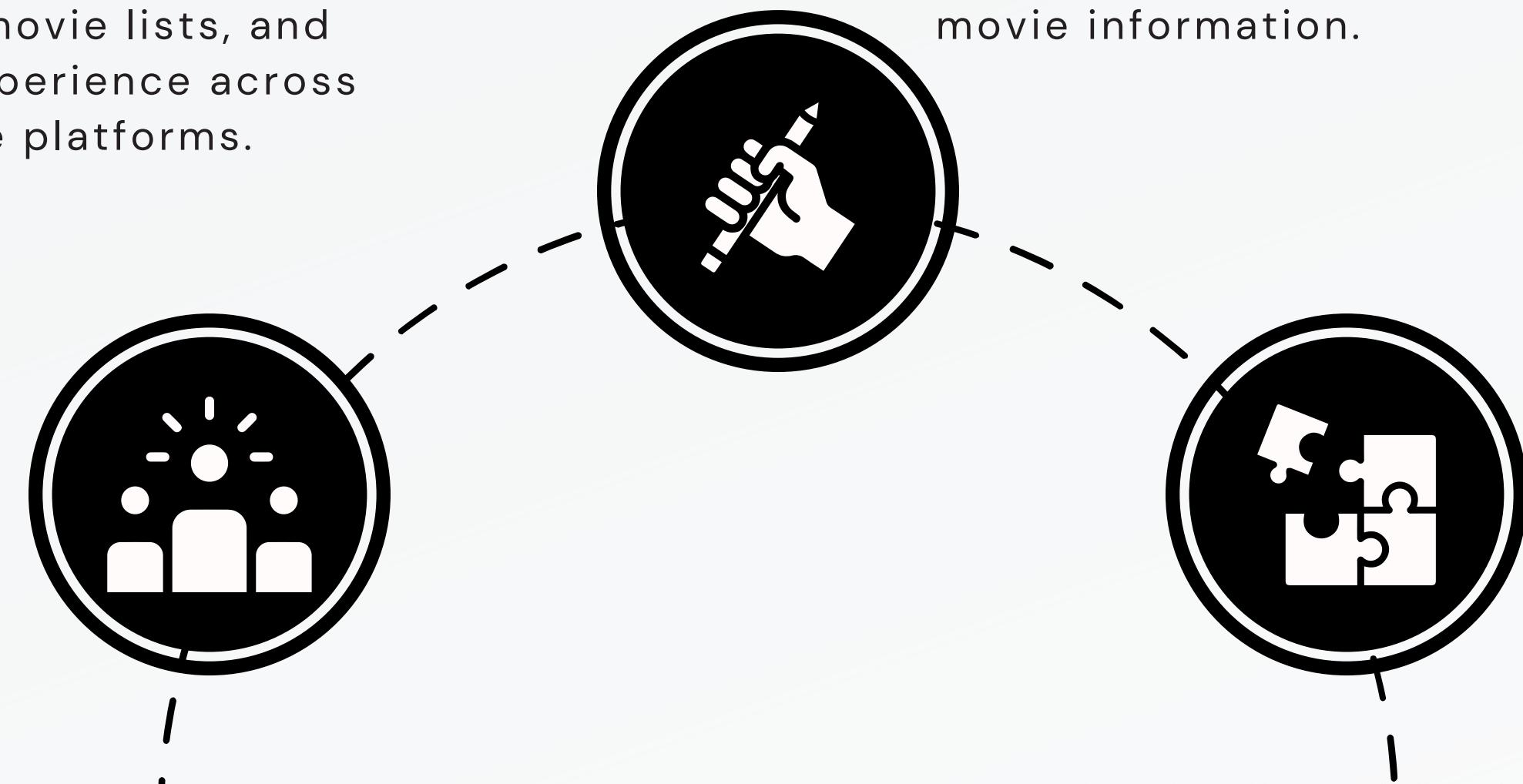
# PROBLEM STATEMENT

## Objective

The main challenge was to create an engaging and user-friendly platform where movie enthusiasts could access detailed information about their favorite films, manage their personal movie lists, and enjoy a seamless user experience across both web and mobile platforms.

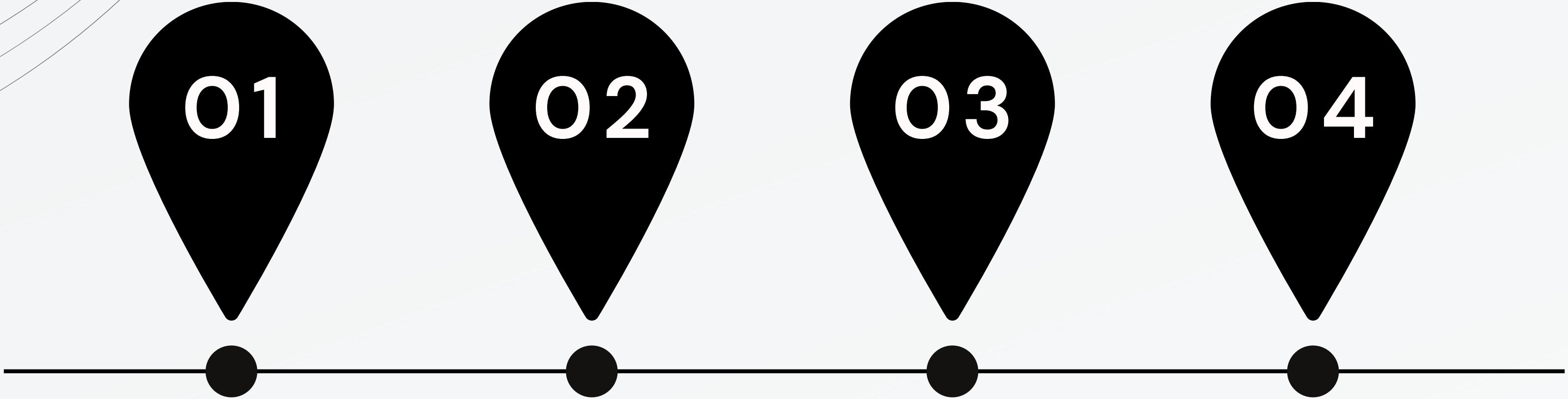
## Key Points

- Lack of a centralized platform for movie information and management.
- Need for a responsive and user-friendly interface.
- Importance of robust backend to handle user data and movie information.



# DEVELOPMENT PHASES

A structured approach divided into clear phases ensured a smooth and efficient development process.



01

02

03

04

## SERVERSIDE

Built a robust API using Node.js, Express.js, and MongoDB, serving as the backbone of the application.

## REACT

Developed a user-friendly and responsive interface using React.

## ANGULAR

Expanded the application with a cross-platform Angular client.

## TESTING

Completed integration, testing, and optimizations to ensure a seamless user experience.

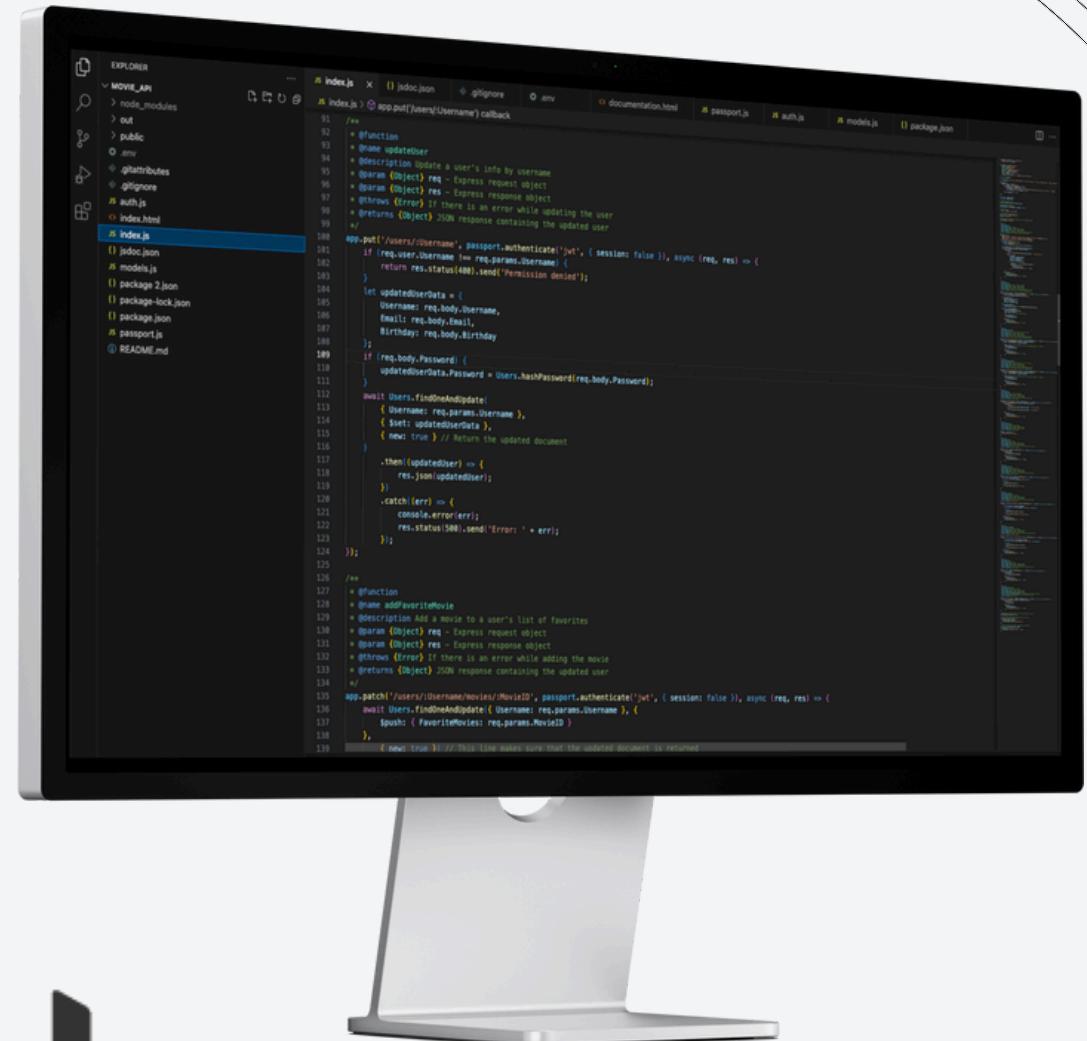
# SERVER-SIDE DEVELOPMENT

## Content:

- **Objective:** Building the server-side infrastructure for the myFlix app.
- **Technologies:** Node.js, Express.js, MongoDB, Mongoose.
- **Core Features:**
  - Return a list of all movies.
  - Return data about a single movie, genre, or director.
  - User registration and profile management.
  - Manage favorite movies list.

## Challenges:

- Integration of security features (e.g., authentication).
- Optimization of API performance.



The image shows a computer monitor displaying a code editor interface. The code editor is showing a file named 'index.js' which contains Node.js server-side code. The code includes imports for express, mongoose, and passport modules. It defines routes for user authentication and profile management, including endpoints for logging in, updating user profiles, and managing favorite movies. The code uses promises and callbacks to handle database operations and return JSON responses.

```
const express = require('express');
const mongoose = require('mongoose');
const passport = require('passport');

// Import routes
const authRoutes = require('./routes/auth');
const movieRoutes = require('./routes/movies');

// Create express app
const app = express();

// Connect to MongoDB
mongoose.connect('mongodb://localhost:27017/myflix', { useNewUrlParser: true });

// Use passport middleware
app.use(passport.initialize());
app.use(passport.session());

// Use routes
app.use('/api/auth', authRoutes);
app.use('/api/movies', movieRoutes);

// Start the server
const PORT = process.env.PORT || 3001;
app.listen(PORT, () => {
  console.log(`Server is running on port ${PORT}`);
});
```



# CLIENT-SIDE DEVELOPMENT (REACT)

## Content:

- **Objective:** Create a user-friendly, responsive single-page application.
- **Technologies:** React, Bootstrap, Parcel.
- **Core Features:**
  - **Main View:** Display all movies, filter by title.
  - **Single Movie View:** Show movie details and manage the favorites list.
  - **Login and Signup Views:** User login and registration.
  - **Profile View:** User management and editing.

## Challenges:

- Implementing a seamless UI/UX.
- Ensuring the application's performance and scalability.



# CLIENT-SIDE DEVELOPMENT (ANGULAR)

## Content:

- **Objective:** Develop an alternative frontend version of the myFlix app using Angular.
- **Technologies:** Angular, TypeScript, SCSS.
- **Core Features:**
  - **Main View:** Display all movies, filter by title.
  - **Single Movie View:** Show movie details and manage the favorites list.
  - **Login and Signup Views:** User login and registration.
  - **Profile View:** User management and editing.

## Challenges:

- Managing different frameworks (React and Angular) and comparing their pros and cons.
- Adapting the code for different platforms.



# SUMMARY AND CONCLUSION



- Successful implementation of both the server-side and client-side.
- Utilization of modern web technologies to create a fully functional app.
- Comparative insights into Angular and React for large web projects.

SUCCESSES N°1



- Handling complex data structures.
- Enhancing teamwork and communication skills.
- Understanding the differences and strengths of Angular vs. React.

LESSONS N°2



- Plans for further optimization and feature enhancements.
- Considerations for scaling the application.
- Potential areas for future development or research.

OUTLOOK N°3

# TESTIMONIALS

## Officer Peanut



The myFlix app completely transformed how I organize and track my favorite movies. The user-friendly design and smooth functionality make it a joy to use every day.



## Sam



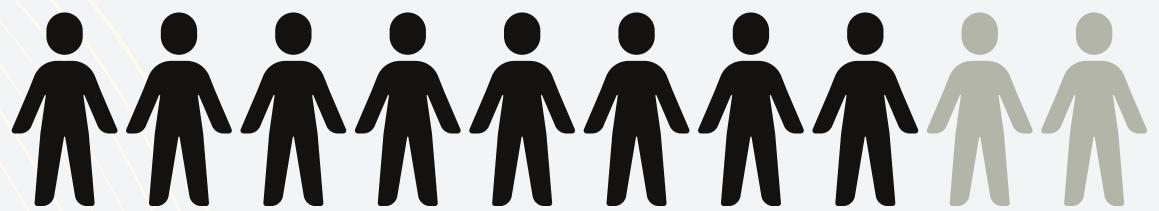
Finally, an app that lets me easily manage my movie collection across all devices! The seamless experience between the web and mobile versions is fantastic. It works even in heaven!



# STATISTICS

95% of users reported finding the user interface intuitive and easy to navigate.

**95%**



# THANK'S FOR WATCHING

*Explore the Live Demos:*

- [myFlix React Demo](#)
- [myFlix Angular Demo](#)

